# MulT: An End-to-End Multitask Learning Transformer

Deblina Bhattacharjee, Tong Zhang, Sabine Süsstrunk and Mathieu Salzmann
School of Computer and Communication Sciences, EPFL, Switzerland
{deblina.bhattacharjee, tong.zhang, sabine.susstrunk, mathieu.salzmann}@epfl.ch

## Abstract

*We propose an end-to-end **Mul**titask Learning **T**ransformer framework, named **MulT**, to simultaneously learn multiple high-level vision tasks, including depth estimation, semantic segmentation, reshading, surface normal estimation, 2D keypoint detection, and edge detection. Based on the Swin transformer model, our framework encodes the input image into a shared representation and makes predictions for each vision task using task-specific transformer-based decoder heads. At the heart of our approach is a shared attention mechanism modeling the dependencies across the tasks. We evaluate our model on several multitask benchmarks, showing that our MulT framework outperforms both the state-of-the art multitask convolutional neural network models and all the respective single task transformer models. Our experiments further highlight the benefits of sharing attention across all the tasks, and demonstrate that our MulT model is robust and generalizes well to new domains. Our project website is at https://ivrl.github.io/MulT/.*

## 1. Introduction

First proposed in [45], transformers have made great strides in a wide range of domains. For instance, previous works [13, 25, 35–37, 52] have demonstrated that transformers trained on large datasets learn strong representations for many downstream language tasks; and models based on transformers have achieved promising results on image classification, object detection, and panoptic segmentation [5, 6, 14, 18, 34, 38, 44, 49, 58]. In contrast to these works that focus on a single task, in this paper, we investigate the use of transformers for multitask learning.

Although a few works have studied the use of transformers to handle multiple *input modalities*, such as images and text, they typically focus on a single *task*, e.g., visual question answering [19, 20, 23, 28], with the exception of [19], which tackles several language tasks but a single vision one. By contrast, our goal is to connect multiple vision tasks covering the 2D, 3D, and semantic domains. To this end we address the following questions: Can a transformer model trained jointly across tasks improve the performance in each task relative to single-task transformers? Can one explicitly encode dependencies across tasks in a transformer-based framework? Can a multitask transformer generalize to unseen domains?

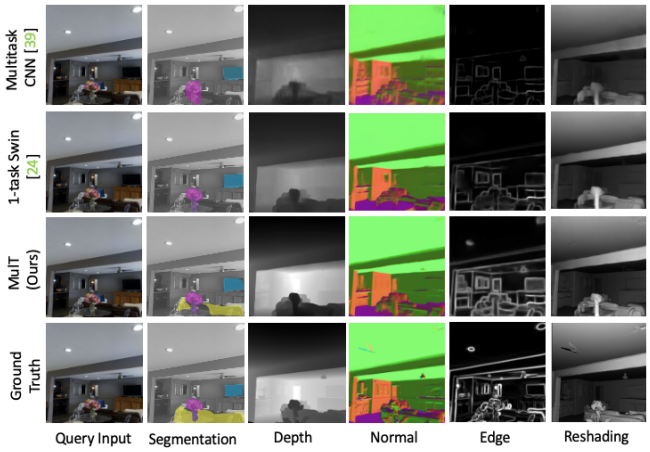


Figure 1. **Motivation for MulT.** Our MulT model, which is a transformer-based encoder-decoder model with shared attention to learn task inter-dependencies, produces better results than both the dedicated 1-task transformer models (1-task Swin [26]) and the state-of-the-art multitask CNN baseline [41].

To the best of our knowledge, only [9, 30, 40] have touched upon the problem of addressing multiple tasks with transformers. However, none of these works aims to encode strong dependencies across the tasks beyond the use of a shared backbone. Furthermore, IPT [9] handles solely low-level vision tasks, such as denoising, super-resolution and deraining, whereas [30] focuses uniquely on the tasks of object detection and semantic segmentation and [40] on scene recognition and importance score prediction in videos. Here, we cover a much wider range of high-level vision tasks and explicitly model their dependencies.

To this end, we introduce MulT, which consists of a transformer-based encoder to transform the input image into a latent representation shared by the tasks, and transformer decoders with task-specific heads producing the final pre-

dictions for each of the tasks. While the MulT encoder mainly utilizes the self-attention mechanism [3, 33] to extract intrinsic features, as most transformers, we equip the decoders with a shared attention mechanism across the different vision tasks, thus allowing the overall framework to encode task dependencies. Thus, we leverage the query and key vectors from the encoder along with the task-specific values in the decoder to predict the task-specific outputs. Our contributions can be summarized as follows:

- We propose an end-to-end multitask transformer architecture that handles multiple high-level vision tasks in a single model.
- We introduce a shared attention between the transformer decoders of the multiple tasks. This shared attention mechanism further improves the performance of each vision task.
- Our framework lets us learn the inter-dependencies across high-level vision tasks.
- We show that our model generalizes and adapts to new domains with a lower average error on the different vision tasks than the existing multitask convolutional models [41, 53].

Our exhaustive experiments and analyses across a variety of tasks show our MulT model not only improves the performance over single-task architectures, but also outperforms the state-of-the-art multitask CNN-based models (as shown in Figure 1) on standard benchmarks, such as Taskonomy [54], Replica [42], NYU [31] and CocoDoom [29] .

## 2. Related Work

**Multitasking.** In its most conventional form, multi-task learning predicts multiple outputs out of a shared encoder/representation for an input [55]. Prior works [24, 41, 43, 53, 54] follow this architecture to jointly learn multiple vision tasks using a CNN. Leveraging this encoder-decoder architecture, IPT [9] was the first transformer-based multitask network aiming to solve low-level vision tasks after fine-tuning a large pre-trained network. This was followed by [30], which jointly addressed the tasks of object detection and semantic segmentation. Recently, [40] used a similar architecture for scene and action understanding and score prediction in videos. However, none of these works connect such a wide range of vision tasks as we do, including 2D, 3D, and semantic domains. Furthermore, they do not explicitly model the dependencies between the tasks, which we achieve via our shared attention mechanism.

**Transformers.** Transformers [45] were originally introduced for language tasks, in particular for machine translation where they showed impressive improvements over recurrent-based encoder-decoder architectures. Since then they have been widely applied to a great range of problems, including speech recognition [16] and language modeling [12, 13]. In the vision domain, transformers have been used to extract visual features, replacing CNNs for object detection, image classification, segmentation and video representation learning [2, 5, 6, 14, 18, 34, 58]. Recently, several works, such as UniT [19] and VILBERT-MT [23], have learned multiple tasks from multimodal domains, such as vision and text. Here, however, we focus on a single input modality: images.

**Learning task inter-dependencies.** Taskonomy [54] studied the relationships between multiple visual tasks for transfer learning and introduced a dataset with 4 million images and corresponding labels for 26 tasks. Following this, a number of recent works have further studied tasks relationships for transfer learning [1, 15, 32, 46]. However, these works differ from multitask learning, in the sense that they analyze a network trained on a source task and applied to a different target task, whereas we study the effect of leveraging multiple tasks during training. In [41], Standley et al. found notable differences between transfer task affinity and multi-task affinity and showed the benefits of leveraging structural similarities between tasks at all levels for multitask learning. In this work, we further study the task inter-dependencies, but by designing a multitask transformer model instead of a CNN one. Our MulT model lets us learn the inter-dependencies across high-level vision tasks and further improves the task inter-dependencies seen in CNN-based models.

**Attention mechanisms.** While there have been a myriad of attention mechanisms [8, 11, 47, 48, 50, 51] to exploit long range dependencies using transformers, none of the prior works utilize a cross-task shared attention for multitask learning. This is what we propose in this work to handle multiple vision tasks.

## 3. MulT: A Multitask Transformer

Our model, MulT, follows the principle of a transformer encoder-decoder architecture [45]. It consists of a transformer-based encoder to map the input image to a latent representation shared by the tasks, followed by transformer decoders with task-specific heads producing the predictions for the respective tasks. Figure 2 shows an overview of our MulT framework. For our transformer-based encoder, we use a pyramidal backbone, named the Swin Transformer [26] to embed the visual features into a list of hidden states that incorporates global contextual information. We then apply the transformer decoders to progressively decode and upsample the tokenized maps from the encoded image. Finally, the representation from the transformer decoder is passed to a task-specific head, such

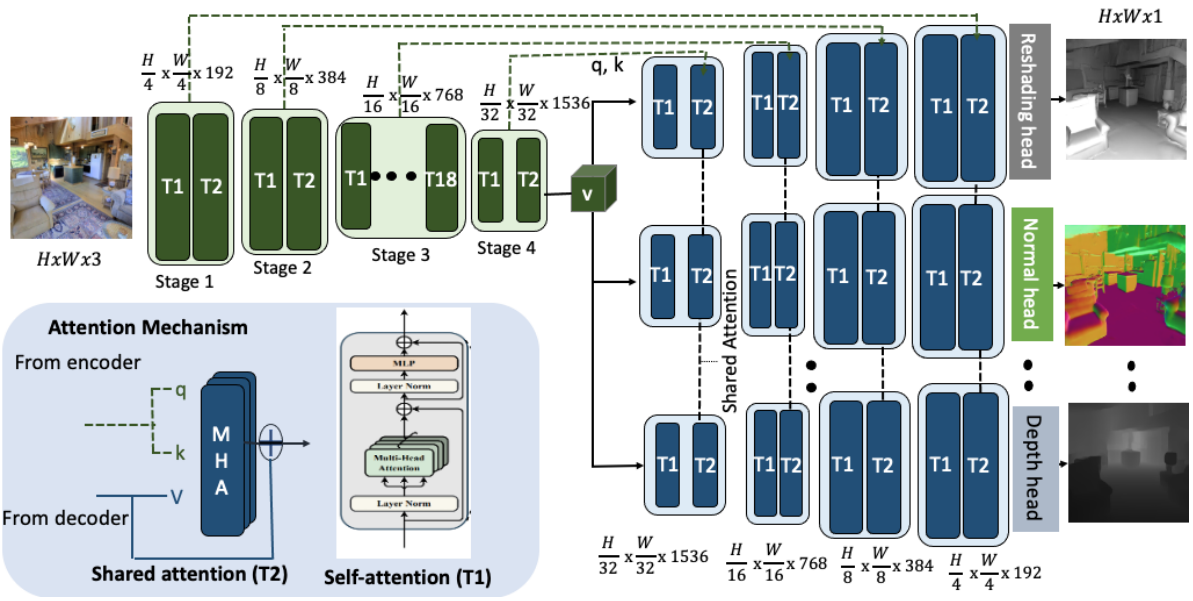


Figure 2. **Detailed overview of our MulT architecture.** Our MulT model builds upon the Swin transformer [26] backbone and models the dependencies between multiple vision tasks via a shared attention mechanism (shown in the bottom left), which we introduce in this work. The encoder module (in green) embeds a shared representation of the input image, which is then decoded by the transformer decoders (in blue) for the respective tasks. Note that the transformer decoders have the same architecture but different task heads. The overall model is jointly trained in a supervised manner using a weighted loss [10] of all the tasks involved. For clarity, only three tasks are depicted here.

as a simple two layer classifier (in the case of segmentation), which outputs the final predictions. Given the simplicity of MulT, it can be extended easily to more tasks. We empirically show that our model can jointly learn 6 different tasks and generalizes well to new domains. The following sections describe the details of each component in MulT.

### 3.1. Encoder Module

For the encoder, we adopt Swin-L [26], which applies stacked transformers to features of gradually decreasing resolution in a pyramidal manner, hence producing hierarchical multi-scale encoded features, as shown in Figure 2. In particular, following the ResNet [17] structure and design rules, four stages are defined in succession: each of them contains a patch embedding step, which reduces the spatial resolution and increases the channel dimension, and a columnar sequence of transformer blocks. The initial basic patch embedding in the first stage is performed with square patches of size $p_H = p_W = 4$ and with channel size C = 192, without the addition of the 'class' token; the patch merging in all three subsequent stages takes the output tokens of the previous stage, reshapes them in a 2D representation and aggregates neighboring tokens in non-overlapping patches of size $p_H = p_W = 2$ through channel-wise concatenation and a linear transformation that halves the resulting number of channels (hence doubles the number of channels with respect to the input tokens). This approach halves the resolution and doubles the channel dimension at every intermediate stage, matching the behavior of typical fully-convolutional backbones and producing a feature pyramid (with output sizes of 1/4, 1/8, 1/16, 1/32 of the original resolution) compatible with most previous architectures for vision tasks.

Following [17], most of the computation is concentrated in the third stage: Out of a total of $N = 24$ transformer encoders, 2 blocks are in the first, second and fourth stage and 18 are in the third stage. In each block, the self-attention is repeated according to the number of heads used and depending on the stage of the encoding process. This is done to match the increase in the channel dimensions, where the dimensions $M = \{6, 12, 24, 48\}$ in the first, second, third and fourth stage, respectively. However, the high resolution in the first two stages does not allow the use of global self-attention, due to its quadratic complexity with respect to the token sequence length. To solve this issue, in all stages, the tokens, that are reshaped in a 2D representation, are divided into non-overlapping square windows of size $h = w = 7$, and the intra-window self-attention is independently computed for each of them. This means that each token attends to only the tokens in its own window, both as a query and as a key/value. A possible downside of this approach could be that the restriction to fixed local windows completely stops any type of global or long-range interaction. The adopted solution is to alternate regular window partitioning with another non-overlapping partitioning in which the windows are shifted by half their size, $\lfloor h/2 \rfloor = \lfloor w/2 \rfloor = 3$, both

in the height and width dimensions. This has the effect of gradually increasing the virtual receptive field of the subsequent attention computations.

### 3.2. Decoder Module

Inspired by the two CNN-based decoders proposed in [56], we develop corresponding conceptually similar transformer-based versions. The general idea is to replace convolutional layers with windowed transformer blocks. Specifically, our decoder architecture consists of four stages, each containing a sequence of 2 transformer blocks for a total of 8. In each stage, the two sequential transformer blocks allow us to leverage inter-window connectivity by alternating regular and shifted window configurations as in the encoder. Between consecutive stages, we use an upsampling layer to double the spatial resolution and half the channel dimension; we therefore adjust the number of attention heads accordingly to 48, 24, 12, 6, in the first, second, third and fourth stage, respectively. The spatial/channel shape of the resulting feature maps matches the outputs of the encoder stages, which are delivered to the corresponding decoder stages by skip connections. This yields an hourglass structure with mirrored encoder-decoder communication: the lower-resolution stages of the decoder are guided by the higher-level deeper encoded features and the higher-resolutions stages of the decoder are guided by the lower-level shallower encoded features, allowing to gradually recover information in a coarse to fine manner and to exploit the different semantic levels where they are more relevant. Note that the first transformer block in each stage of the decoder uses a regular window partitioning while the second uses a shifted window partitioning; this can easily be extended to using a longer sequence of transformer blocks, as long as the length is a multiple of 2, which makes it possible to alternate between the two configurations.

To perform multitask prediction, we share the encoder across all tasks and use task-specific decoders with the same architecture but different parameter values. We then simply append task-specific heads to the decoder. For instance, a model jointly trained for semantic segmentation and depth prediction will have two task-specific heads: one predicting $K$ channels followed by a softmax for semantic segmentation and one predicting a single channel followed by a sigmoid for depth estimation.

### 3.3. Shared Attention

To account for the task dependencies beyond sharing encoder parameters, we develop a shared attention mechanism that integrates the information contained in the encoded features into the decoding stream. Let us now describe how this mechanism works for one particular decoder stage. Note that we apply the same procedure for all decoder stages.

Formally, for one task $t$ and one particular decoder stage, let $x^t$ denote the upsampled output of the previous stage, and $x_{sa}$ the output of the encoding stage operating at the same resolution. As illustrated in Figure 3, the decoder stage takes both $x^t$ and $x_{sa}$ as input. The standard way to compute self-attention for task $t$ would be to obtain the key, query and value vectors from its own decoder output $x^t$ only. By contrast, for our shared attention, we use only one of the task streams to calculate the attention. That is, we compute a query $q^r_{sa}$ and a key $k^r_{sa}$ from $x_{sa}$ (coming from the encoder) by using the linear layers, shown in Figure 3, of the decoder of one particular reference task $r$. To nonetheless reflect the fact that the output of the decoder for task $t$ should be related to this particular task, we compute the values $v^t$ using the previous stage output $x^t$ for task $t$. Thus, we compute attention values from the reference task $r$ as

$$A^r_{sa} = \text{softmax}\left(\frac{q^r_{sa}.{k^r_{sa}}^T}{\sqrt{C^r_{qkv}}} + B^r\right), \tag{1}$$

where $C^r$ is the number of channels and $B^r$ is the bias. For any task $t$, we then compute $\tilde{x}^t = A^r_{sa} v^t$. This $\tilde{x}^t$ is then used by the self-attention head $\text{head}^t_i(.,.)$ to compute $\text{head}^t_i(\tilde{x}^t_i, W^t_i) = \tilde{x}^t_i \cdot W^t_i$, where $W^t_i$ is the learnt attention weight for task $t$ and $\tilde{x}^t_i$ is the i$^{th}$ channel, respectively. Note that this formulation represents the $i^{th}$ instance of the self-attention, which is repeated $M$ times to obtain a multi-head attention as $\text{MHA}^t(.,.)$ for task $t$. Following which, we compute $x^t_{linear}$ by linearly projecting the output of $\text{MHA}^t(.,.)$. Finally, we obtain $y^t$ as follows:

$$\begin{aligned} &\text{MHA}^t(\tilde{x}^t, W) = \text{Concat}(\text{head}^t_1, \cdots, \text{head}^t_M)\mathbf{W}\ , \\ &x^t_{linear} = \text{MHA}^t(.,.)\ , \\ &y^t = x^t + x^t_{linear}\ , \end{aligned} \tag{2}$$

where $\mathbf{W}$ indicates the multi-head attention weight. Empirically, we have found that the attention from the surface normal task stream benefits our 6-task MulT model, and we thus take this task as reference task $r$, whose attention is shared across the tasks. As shown in Figure 3, $x^r$ is the upsampled output of the previous stage of a particular decoder for the reference task, taken here as surface normal prediction.

Note that our shared attention differs from the co-attention introduced in prior works [7], where the value and key are passed via a skip connection from the encoder layers. Figure 4 shows the effect of adding our shared attention mechanism across the tasks, where our MulT with the shared attention mechanism improves the results across all the tasks in comparison with our MulT model without the shared attention.

**Task Heads and Loss.** The feature maps from the transformer decoder modules are input to different task-specific

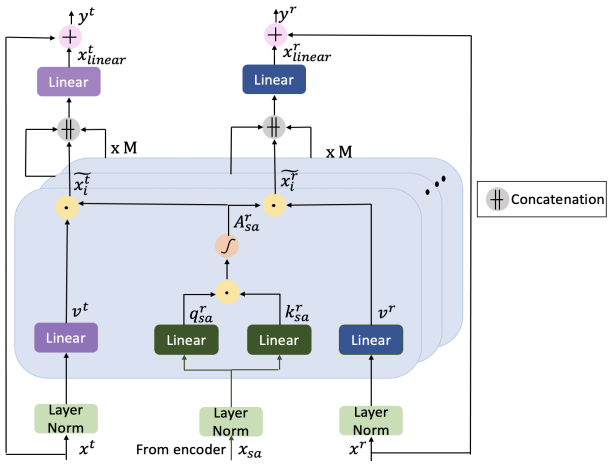


Figure 3. Overview of our **shared attention** mechanism.

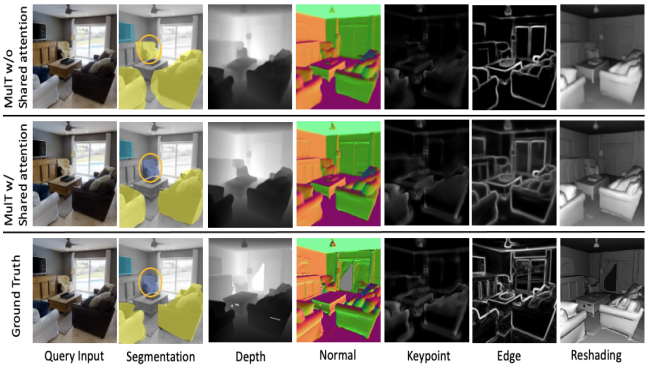


Figure 4. **Motivation of the shared attention mechanism on our MulT model.** The shared attention mechanism learns the task inter-dependencies and improves the prediction for each task. For instance, the yellow circled region shows how our MulT model with shared attention across tasks improves the semantic segmentation performance, where the chair mask is correctly classified in our predictions as in the ground truth. However in the MulT model without the shared attention, the chair is predicted as a couch mask. Best viewed on screen and when zoomed in.

heads to make subsequent predictions. Each class head includes a single linear layer to output a $H \times W \times 1$ map, where $H, W$ are the input image dimensions. We employ a weighted sum [10] based task-specific losses to jointly train the network, where the losses are calculated between the ground truth and final predictions for each task. In particular, we use cross-entropy for segmentation, rotate loss [54] for depth, and $L1$ loss for surface normals, 2D keypoints, 2D edges and reshading, respectively. Note that we employ these losses to maintain consistency with the baselines [41, 53, 54].

# 4. Experiments and Results

To provide a thorough analysis of MulT and also compare it with well-established prior work, we experiment with jointly learning prominent, high-level vision tasks.

## 4.1. Datasets

We evaluate MulT using the following datasets:

**Taskonomy [54]** is used as our main training dataset. It comprises 4 million real images of indoor scenes with multi-task annotations for each image. The experiments were performed using the following 6 tasks: *semantic segmentation* $(\mathcal{S})$*, depth (zbuffer)* $(\mathcal{D})$*, surface normals* $(\mathcal{N})$*, 2D keypoints* $(\mathcal{K})$*, 2D (Sobel) texture edges (E) and reshading* $(\mathcal{R})$. The tasks were selected to cover 2D, 3D, and semantic domains and have sensor-based/semantic ground truth. We report results on the Taskonomy test set.

**Replica [42]** comprises high-resolution 3D ground truth and enables more reliable evaluations of fine-grained details. We test all the networks on 1227 images from Replica (with and without fine-tuning).

**NYU [31]** comprises 1449 images from 464 different indoor scenes. We test all the networks on NYU (with and without fine-tuning).

**CocoDoom [29]** contains synthetic images from the Doom video game. We use it as an out-of-training-distribution dataset.

## 4.2. Training Details

We jointly train MulT on multiple tasks, including, semantic segmentation, depth estimation, 2D keypoint detection, 2D edge detection, surface normal estimation and reshading. In our implementation, we train with a batch size of 32 on 32 Nvidia V100-SXM2-32GB GPUs in a distributed fashion, using PyTorch. We use the weighted Adam optimizer [27] with a learning rate of 5e-5 and the warm-up cosine learning rate schedule (using 2000 warm-up iterations). The optimizer updates the model parameters based on gradients from the task losses.

## 4.3. Baselines

We compare our MulT model with the following state-of-the-art baselines.

**Baseline UNet (for single-task or independent learning)** constitutes our CNN-based baseline. We use it as a reference for all the multitask models.

**Baseline Swin transformer [26] (for single-task or independent learning)** constitutes the single task transformer baseline. It is almost identical to our MulT model, except for not including shared attention and for being trained with only one dedicated task. We use it to evaluate the benefits of our multitask learning strategy

**Multi-task learning [22] (MTL)** comprises a network with one shared encoder and multiple decoders each dedicated to a task. This baseline further identifies if tasks are inter-dependent, such that a shared representation can give

comparable performance across multiple tasks, without explicitly adding task constraints.

**Taskonomy [54]** studies the relationships between multiple visual tasks for transfer learning.

**Taskgrouping [41]** studies task compatibility in multitask learning, thus providing a framework for determining which tasks should be trained jointly and which tasks should be trained separately.

**Cross-task consistency [53]** presents a general and data-driven framework for augmenting standard supervised learning with cross-task consistency. It is inspired from Taskonomy [54] but adds a consistency constraint to learn multiple tasks jointly.

Note that we do not compare our method with the contemporary work [19] as it focuses on *bimodal* multitask learning for vision- and language-related tasks. By contrast, in this work, we tackle *unimodal* multitask learning for high-level vision tasks. All the multitask baselines were trained using their best model configurations as in [22, 41, 53, 54], respectively.

| | Relative Performance On | | | | | |
|---|---|---|---|---|---|---|
| | $\mathcal{S}$ | $\mathcal{D}$ | $\mathcal{N}$ | $\mathcal{K}$ | $E$ | $\mathcal{R}$ |
| $\mathcal{S}$ | - | +3.83% | -1.42% | -1.33% | +33.9% | -0.80% |
| $\mathcal{D}$ | +4.83% | - | +2.77% | -1.92% | +35.2% | +3.93% |
| $\mathcal{N}$ | +11.3% | +8.35% | - | +91.2% | +77.1% | +9.09% |
| $\mathcal{K}$ | +5.11% | +0.57% | -6.88% | - | +70.1% | +0.21% |
| $E$ | +6.09% | +4.33% | -0.73% | +4.75% | - | +5.11% |
| $\mathcal{R}$ | +8.61% | +4.45% | +5.91% | +1.95% | +33.9% | - |

Table 1. **Quantitative comparison of our MulT model with a single-task dedicated Swin transformer baseline [26].** Our MulT model is jointly trained in a pairwise manner on the Taskonomy benchmark [54]. For instance, in the first row, second column, we show the results of our MulT model trained with semantic segmentation and depth in a pairwise manner, and tested on the task of depth estimation. The relative performance percentage for each task is evaluated by taking the percentage increase or decrease w.r.t. the single-task baseline. The results here are reported on the Taskonomy test set. The columns show the task tested on, and the rows show the other task used for training.

### 4.4. Quantitative Results

The results in Table 1 show the relative performance of our MulT model when trained on pairs of tasks and tested on one of the two tasks. We observe that, out of the pairwise-trained multitask models, surface normals help the other vision tasks. However, the performance of normals tends to decrease w.r.t. its single task dedicated model, except when used in conjunction with either depth predition or reshading. Note that the trends we observe are similar to those shown in [41] for the CNN case. This suggests that transformers follow a similar behavior to that of CNNs in the presence of multiple tasks.

In cases of more than two tasks, we observe, as in [41], that effectively leveraging between 3 and 6 tasks required increasing the size of the decoder modules. Altogether, reporting results for all possible task combinations requires training $(2^6 - 1)$ models. Here, we focus on the 6-task case, but provide 3-task, 4-task, and 5-task results in the supplementary material. The results of our 6-task MulT model and of the baselines are reported in Table 2 and Table 3 for the Taskonomy test set [54], and the Replica [42] and NYU [31] dataset, respectively. Our MulT model outperforms the multitask CNN baselines as well as the 1-task CNN and Swin ones. Furthermore, as can be verified from the results in the supplementary material, increasing the number of tasks improves the results of our MulT model, e.g., a 6-task network outperforms a 5-task one, which in turn outperforms a 4-task network.

### 4.5. Qualitative Results

We qualitatively compare the results of our MulT model with different CNN-based multitask baselines [22, 41, 53, 54], as well as with the single task dedicated Swin transformer [26]. The results in Figure 5 show the performance of the different networks on all six vision tasks. All the multitasking models are jointly trained on the six tasks on the Taskonomy benchmark, and the single task dedicated Swin models are trained on the respective tasks. Our MulT model yields higher-quality predictions than both the single task Swin baselines and the multitask CNN baselines. We provide additional qualitative results in the supplementary material.

### 4.6. Generalization to New Domains

In this section, we demonstrate how well MulT generalizes to new domains without any fine-tuning, and how efficiently MulT can adapt to a new domain by fine-tuning on a small set of training examples from the new domain. To this end, we compare our MulT model and the two baselines of Taskgrouping (TG) [41] and Cross-task consistency (CT) [53] on two new domains, namely, Gaussian-blurred images from Taskonomy [21] and images from the Cocodoom [29] dataset. Note that all the networks were trained on the vanilla Taskonomy dataset [54]. When fine-tuning the networks, we use either 16 or 128 images from the new domain. The original training data (Taskonomy) is retained during fine-tuning to prevent the networks from forgetting the original domain.

The results in Table 4 and Figure 6 show that our MulT model yields better generalization and adaptation to new domains, both with and without fine-tuning. These findings confirm the observations made in [4] for the single-task scenario. The cross-task consistency [53] model shows improved performance in comparison to the Taskgrouping [41] baseline because of its explicitly enforced consistency constraint, whereas the Taskgrouping model [41] suffers due to the joint task pairings and the lack of an attention

| | Relative Performance On Taskonomy Test Set [54] | | | | | |
|---|---|---|---|---|---|---|
| | $\mathcal{S}$ | $\mathcal{D}$ | $\mathcal{N}$ | $\mathcal{K}$ | $E$ | $\mathcal{R}$ |
| MTL [22] vs 1-task CNN [39] | +2.05% | +3.11% | +4.38% | -1.29 % | +45.22% | +2.99% |
| Taskonomy [54] vs 1-task CNN [39] | +2.63% | -3.82 % | +2.95% | +10.13 % | +59.05% | +4.52% |
| Taskgrouping [41] vs 1-task CNN [39] | +6.24% | +3.36% | +4.23% | +21.77% | +73.6 % | +5.79% |
| Cross-task [53] vs 1-task CNN [39] | +9.01% | +6.77% | +5.61% | +23.20% | +75.8 % | +11.1% |
| **MulT** vs 1-task Swin [26] | +19.7% | +10.2% | +8.72% | +94.75% | +88.8% | +16.4% |
| **MulT** vs 1-task CNN [39] | **+21.6%** | **+11.5%** | **+9.71%** | **+97.04%** | **+92.9%** | **+21.0%** |

Table 2. **Quantitative comparison of our MulT model with baselines when jointly trained for six tasks on the Taskonomy benchmark [54].** Our six-task MulT model consistently outperforms all the baselines, including the multitasking CNN baselines and the single-task CNN and Swin baselines. The relative performance percentage for each task is evaluated by taking the percentage increase or decrease w.r.t. the single-task baseline. The results here are reported on the Taskonomy test set. Bold and underlined values show the best and second-best results, respectively.

| | Relative Performance On | | | | |
|---|---|---|---|---|---|
| | Replica Dataset [42] | | | NYU Dataset [31] | |
| | $\mathcal{D}$ | $\mathcal{N}$ | $\mathcal{R}$ | $\mathcal{S}$ | $\mathcal{D}$ |
| MTL [22] vs 1-task CNN [39] | +2.53% | +3.03% | +1.87% | +1.13% | +2.72% |
| Taskonomy [54] vs 1-task CNN [39] | -4.55% | +1.99% | +3.33% | +2.05% | -4.07% |
| Taskgrouping [41] vs 1-task CNN [39] | +2.75% | +4.09% | +5.47% | +6.01% | +2.91% |
| Cross-task [53] vs 1-task CNN [39] | +5.10% | +4.33% | +9.55% | +8.10% | +5.71% |
| **MulT** vs 1-task Swin [26] | +8.33% | +7.05% | +14.2% | +13.3% | +8.54% |
| **MulT** vs 1-task CNN [39] | **+10.1%** | **+8.59%** | **+19.6%** | **+15.7%** | **+10.4%** |

Table 3. **Quantitative comparison of our MulT model with baselines on the Replica benchmark and the NYU benchmark.** We apply our MulT model, jointly trained on 6 tasks on the Taskonomy dataset, to test the depth, normals and reshading prediction performances on the Replica dataset [42], and the segmentation and depth prediction performance on the NYU dataset [31]. Our six-task MulT model consistently outperforms all the baselines, including the multitasking CNN baselines and the single-task CNN and Swin baselines. The relative performance percentage for each task is evaluated by taking the percentage increase or decrease w.r.t. the single-task baseline. Bold and underlined values show the best and second-best results, respectively.

mechanism or an additional constraint. Nevertheless, our MulT model outperforms both these baselines and shows better generalization.

| | Generalization to New Domains | | | | | | |
|---|---|---|---|---|---|---|---|
| | No. of | Error (w/ Fine-tuning)↓ | | | Error (w/o Fine-tuning)↓ | | |
| Domains | images | MulT | CT [53] | TG [41] | MulT | CT [53] | TG [41] |
| Blur [21] | 128 | **12.6** | 17.4 | 21.9 | **27.0** | 46.2 | 55.1 |
| (Taskonomy) | 16 | 17.5 | 22.2 | 26.3 | | | |
| CocoDoom | 128 | **13.3** | 18.5 | 25.3 | **39.3** | 54.3 | 67.7 |
| [29] | 16 | 20.9 | 27.1 | 39.9 | | | |

Table 4. **Domain generalization on Taskonomy blur data [21] and CocoDoom [29].** Our MulT model shows better abilities to generalize and adapt to new domains, both with and without fine-tuning. Bold and underlined values show the best and second-best results, respectively.

**Supplementary Material.** We defer additional discussions and experiments, particularly analyzing the effect of the shared attention in our MulT model and the effect of the network size for different task combinations, as well as additional qualitative results to the supplementary material. We also analyze the number of parameters required by each model and the environmental impact of training such models in the supplementary material.

## 5. Conclusion and Limitations

In this work, we have shown that the transformer framework can be applied to jointly handle multiple tasks within a single end-to-end encoder-decoder framework. Our MulT model simultaneously addresses 6 different vision tasks, learning them in a single training step and outperforming an independent single task model on each task with a compact set of shared parameters. This allows us to use a single network to handle multiple vision tasks instead of multiple single task networks, thereby reducing the computational cost, for both training and inference. Furthermore, our MulT model outperforms the state-of-the-art CNN-based multitasking models, in terms of both performance in the original domain and generalization/adaptation to new domains.

Our current framework nonetheless suffers from some limitations:

**Data dependency.** Although we validated our findings using various architectures and benchmarks, the results of our approach, as any deep learning one, are in principle data specific. In particular, MulT is a data intensive architecture, and thus when trained on a limited amount of data, it may not achieve the same performance as reported in this work. Note, however, that this is also the case for both single task transformers and the CNN-based multitask baselines.

**Unpaired Data.** Our current framework, as the CNN-based multitask baselines, requires paired training data. Extending our approach to unlabeled/unpaired data, as in [57], appears feasible and remains open for future work.

**Modeling efficient attention.** Our current framework makes use of shared attention across the visual tasks. Ex-

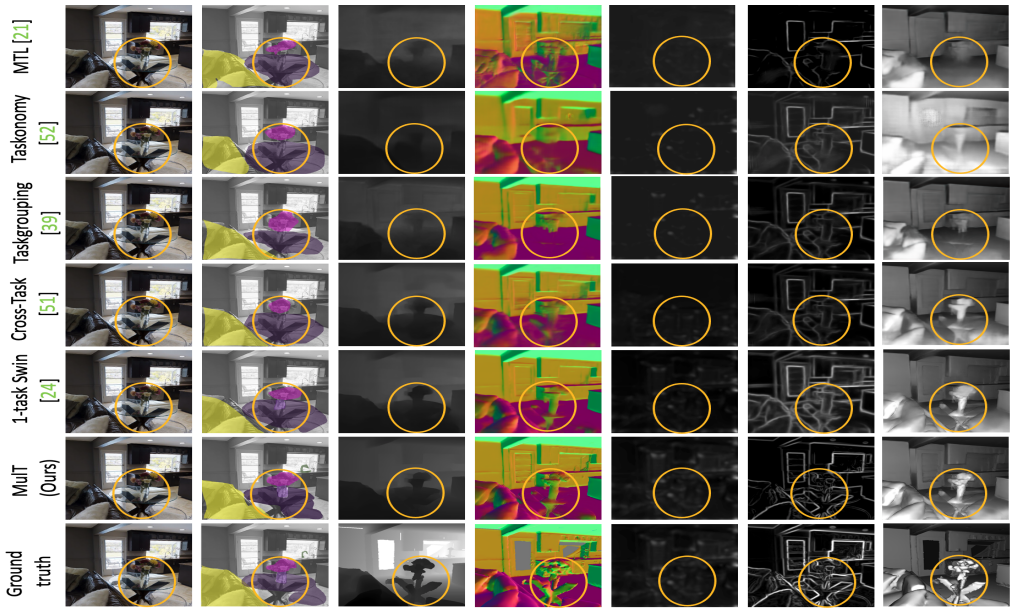


Figure 5. **Qualitative comparison on the six vision tasks** of the Taskonomy benchmark [54]. From top to bottom, we show qualitative results using MTL [22], Taskonomy [54], Taskgrouping [41], Cross-task consistency [53], the single-task dedicated Swin transformer [26] and our six-task **MulT** model. We show, from left to right, the input image, the semantic segmentation results, the depth predictions, the surface normal estimations, the 2D keypoint detections, the 2D edge detections and the reshading results for all the models. All models are jointly trained on the six vision tasks, except for the Swin transformer baseline, which is trained on the independent single tasks. Our MulT model outperforms both the single task Swin baselines and the multitask CNN based baselines. Best seen on screen and zoomed within the yellow circled regions.

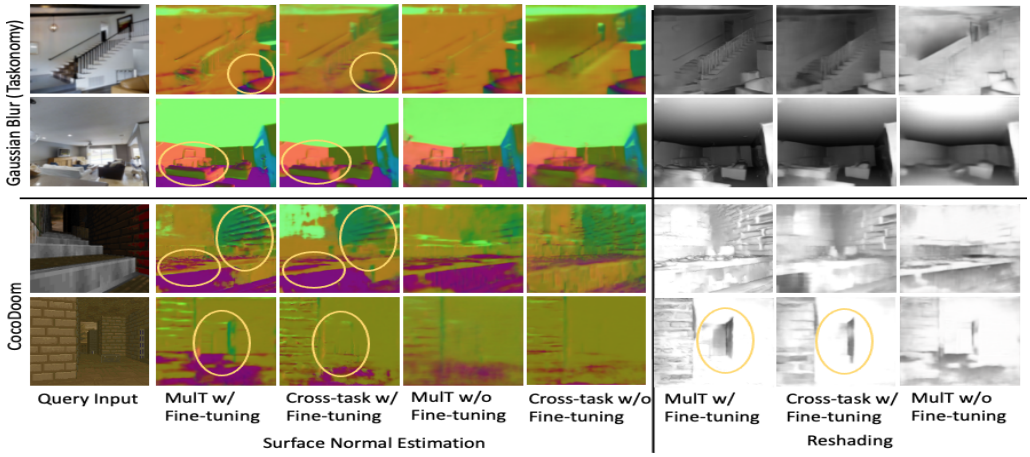


Figure 6. **Generalization to new domains.** Our MulT model generalizes better to new domains than the Cross-task [53] baseline, both when fine-tuned and not fine-tuned, across the tasks of surface normal prediction and reshading. This shows the benefits of our shared attention module. We test the models on two target domains, Gaussian blur applied to the Taskonomy images [54] and the out-of-distribution CocoDoom dataset [29]. Best viewed on screen and when zoomed in the yellow circled regions.

tending this concept to incorporate local versus global attention, as in [51], appears feasible and remains open for future work.

Besides addressing these limitations, in the future, we plan to extend our methodology to learning different types of tasks like edge occlusions, principal curvatures and unsupervised segmentation, and doing zero-shot learning on new tasks. In addition, it would be worthwhile to explore the robustness of large-scale multitask transformers to adversarial tasks, which could become increasingly problematic as the number and variety of tasks grow.

**Acknowledgement.** This work was supported in part by the Swiss National Science Foundation via the Sinergia grant CRSII5−180359.

## References


[1] Alessandro Achille, Michael Lam, Rahul Tewari, Avinash Ravichandran, Subhransu Maji, Charless Fowlkes, Stefano Soatto, and Pietro Perona. Task2vec: Task embedding for meta-learning. *arXiv:1902.03545, cs.LG*, 2019. 2

[2] Farhat Afza, Muhammad Sharif, Muhammad Attique Khan, Usman Tariq, Hwan-Seung Yong, and Jaehyuk Cha. Multiclass skin lesion classification using hybrid deep features selection and extreme learning machine. *Sensors*, 22(3), 2022. 2

[3] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv: 1409.0473, cs.CV*, 2016. 2

[4] Yutong Bai, Jieru Mei, Alan Yuille, and Cihang Xie. Are transformers more robust than cnns? *arXiv: 2111.05464, cs.CV*, 2021. 6

[5] Irwan Bello, Barret Zoph, Ashish Vaswani, Jonathon Shlens, and Quoc V. Le. Attention augmented convolutional networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. 1, 2

[6] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. *arXiv: 2005.1287, cs.CV*, 2020. 1, 2

[7] Hila Chefer, Shir Gur, and Lior Wolf. Transformer interpretability beyond attention visualization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 782–791, June 2021. 4

[8] Chun-Fu Chen, Rameswar Panda, and Quanfu Fan. Regionvit: Regional-to-local attention for vision transformers. *arXiv:2106.02689, cs.CV*, 2021. 2

[9] Hanting Chen, Yunhe Wang, Tianyu Guo, Chang Xu, Yiping Deng, Zhenhua Liu, Siwei Ma, Chunjing Xu, Chao Xu, and Wen Gao. Pre-trained image processing transformer. *arXiv: 2012.00364, cs.CV*, 2021. 1, 2

[10] Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. GradNorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 794–803. PMLR, 2018. 3, 5

[11] Xiangxiang Chu, Zhi Tian, Yuqing Wang, Bo Zhang, Haibing Ren, Xiaolin Wei, Huaxia Xia, and Chunhua Shen. Twins: Revisiting the design of spatial attention in vision transformers. In *NeurIPS 2021*, 2021. 2

[12] Zihang Dai*, Zhilin Yang*, Yiming Yang, William W. Cohen, Jaime Carbonell, Quoc V. Le, and Ruslan Salakhutdinov. Transformer-XL: Language modeling with longer-term dependency. *arXiv, cs.CV*, 2019. 2

[13] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv:1810.04805, cs.CL*, 2019. 1, 2

[14] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. 1, 2

[15] Kshitij Dwivedi and Gemma Roig. Representation similarity analysis for efficient task taxonomy & transfer learning. *arXiv:1904.11740, cs.CV*, 2019. 2

[16] Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, and Ruoming Pang. Conformer: Convolution-augmented transformer for speech recognition. *arXiv:2005.08100, eess.AS*, 2020. 2

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *arXiv:1512.03385*, 2015. 3

[18] Jie Hu, Li Shen, Samuel Albanie, Gang Sun, and Andrea Vedaldi. Gather-excite: Exploiting feature context in convolutional neural networks. In *Proceedings of NeurIPS*, 2018. 1, 2

[19] Ronghang Hu and Amanpreet Singh. Unit: Multimodal multitask learning with a unified transformer. *arXiv: 2102.10772, cs.CV*, 2021. 1, 2, 6

[20] Ronghang Hu, Amanpreet Singh, Trevor Darrell, and Marcus Rohrbach. Iterative answer prediction with pointer-augmented multimodal transformers for textvqa. *CoRR*, abs/1911.06258, 2019. 1

[21] Jason Jo and Yoshua Bengio. Measuring the tendency of cnns to learn surface statistical regularities. *arXiv:1711.11561, cs.LG*, 2017. 6, 7

[22] Iasonas Kokkinos. Ubernet: Training a 'universal' convolutional neural network for low-, mid-, and high-level vision using diverse datasets and limited memory. *arXiv: 1609.02132, cs.CV*, 2016. 5, 6, 7, 8

[23] Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. Visualbert: A simple and performant baseline for vision and language. *arXiv: 1908.03557, cs.CV*, 2019. 1, 2

[24] Shikun Liu, Edward Johns, and Andrew J. Davison. End-to-end multi-task learning with attention. *CoRR*, abs/1803.10704, 2018. 2

[25] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019. 1

[26] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. *arXiv: 2103.14030, cs.CV*, 2021. 1, 2, 3, 5, 6, 7, 8

[27] Ilya Loshchilov and Frank Hutter. Fixing weight decay regularization in adam. *Openreview*, 2018. 5

[28] Jiasen Lu, Vedanuj Goswami, Marcus Rohrbach, Devi Parikh, and Stefan Lee. 12-in-1: Multi-task vision and language representation learning. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 1

[29] A. Mahendran, H. Bilen, J. F. Henriques, and A. Vedaldi. Researchdoom and cocodoom: Learning computer vision with games, copyright(c) 2014, coco api. *arXiv: 1610.02431, cs.CV*, 2016. 2, 5, 6, 7, 8

[30] Eslam Mohamed and Ahmed El-Sallab. Spatio-temporal multi-task learning transformer for joint moving object detection and segmentation. *arXiv: 2106.11401, cs.CV*, 2021. 1, 2

[31] Pushmeet Kohli Nathan Silberman, Derek Hoiem and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV, CC-BY 4.0*, 2012. 2, 5, 6, 7

[32] Arghya Pal and Vineeth N. Balasubramanian. Zero-shot task transfer. *CoRR*, abs/1903.01092, 2019. 2

[33] Ankur Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. A decomposable attention model for natural language inference. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2249–2255. Association for Computational Linguistics, 2016. 2

[34] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image transformer. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4055–4064. PMLR, 2018. 1, 2

[35] Alec Radford and Ilya Sutskever. Improving language understanding by generative pre-training. In *arxiv*, 2018. 1

[36] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019. 1

[37] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020. 1

[38] Prajit Ramachandran, Niki Parmar, Ashish Vaswani, Irwan Bello, Anselm Levskaya, and Jonathon Shlens. Stand-alone self-attention in vision models. *arXiv: 1906.05909, cs.CV*, 2019. 1

[39] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *arXiv:1505.04597, cs.CV*, 2015. 7

[40] Hongje Seong, Junhyuk Hyun, and Euntai Kim. Video multitask transformer network. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pages 1553–1561, 2019. 1, 2

[41] Trevor Standley, Amir R. Zamir, Dawn Chen, Leonidas Guibas, Jitendra Malik, and Silvio Savarese. Which tasks should be learned together in multi-task learning? *arXiv: 1905.07553, cs.CV*, 2019. 1, 2, 5, 6, 7, 8

[42] Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, and Erik Wijmans et. al. The Replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797, CC-BY 4.0*, 2019. 2, 5, 6, 7

[43] Gjorgji Strezoski, Nanne van Noord, and Marcel Worring. Many task learning with task routing. *arXiv:1903.12117*, 2019. 2

[44] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Herve Jegou. Training data-efficient image transformers; distillation through attention. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139, pages 10347–10357. PMLR, 2021. 1

[45] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017. 1, 2

[46] Aria Wang, Michael Tarr, and Leila Wehbe. Neural taskonomy: Inferring the similarity of task-derived representations from brain activity. In *Advances in Neural Information Processing Systems*, volume 32, 2019. 2

[47] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pvtv2: Improved baselines with pyramid vision transformer. *arXiv:2106.13797, cs.CV*, 2021. 2

[48] Wenxiao Wang, Lu Yao, Long Chen, Deng Cai, Xiaofei He, and Wei Liu. Crossformer: A versatile vision transformer based on cross-scale attention. *arXiv:2108.00154, cs.CV*, 2021. 2

[49] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 1

[50] Weijian Xu, Yifan Xu, Tyler Chang, and Zhuowen Tu. Co-scale conv-attentional image transformers. *arXiv:2104.06399, cs.CV*, 2021. 2

[51] Jianwei Yang, Chunyuan Li, Pengchuan Zhang, Xiyang Dai, Bin Xiao, Lu Yuan, and Jianfeng Gao. Focal self-attention for local-global interactions in vision transformers. *arXiv: 2107.00641, cs.CV*, 2021. 2, 8

[52] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. 1

[53] Amir Zamir, Alexander Sax, Teresa Yeo, Oğuzhan Kar, Nikhil Cheerla, Rohan Suri, Zhangjie Cao, Jitendra Malik, and Leonidas Guibas. Robust learning through cross-task consistency. *arXiv*, 2020. 2, 5, 6, 7, 8

[54] Amir R. Zamir, Alexander Sax, William B. Shen, Leonidas J. Guibas, Jitendra Malik, and Silvio Savarese. Taskonomy: Disentangling task transfer learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE (MIT License Copyright (c) 2017 Stanford Vision and Learning Group), 2018. 2, 5, 6, 7, 8

[55] Yu Zhang and Qiang Yang. A survey on multi-task learning. *arXiv:1707.08114, cs.LG*, 2021. 2

[56] Sixiao Zheng, Jiachen Lu, Hengshuang Zhao, Xiatian Zhu, Zekun Luo, Yabiao Wang, Yanwei Fu, Jianfeng Feng, Tao Xiang, Philip H.S. Torr, and Li Zhang. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In *CVPR*, 2021. 4

[57] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-

consistent adversarial networks. *arXiv: 1703.10593, cs.CV*, 2020. 7

[58] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv: 2010.04159, cs.CV*, 2021. 1, 2